Final Project Report
CPE 232 Data Models

"ModchelinGuide"
The ranking and improvement suggestion system for restaurants in a selected area

Presented By

1.	Karun	Tancharoen	66070503407
2.	Chanya	Kittichai	66070503412
3.	Nuttadon	Lertritdacha	66070503419
4.	Punyanuch	Kajornchaikitti	66070503438
5.	Vitawat	Kitipatthavorn	66070503482
6.	Patiharn	Liangkobkit	66070503489
7.	Nicolas Hermann	Teak	67540460064

Presented To
Dr. Sansiri Tarnpradab

Semester 2/2023
Department of Computer Engineering
King Mongkut's University of Technology Thonburi

Table of Contents

Introduction

Many restaurant owners aim to improve their restaurants to attract more customers. However, they may not be able to address the aspects that significantly impact customer satisfaction and fail. To resolve this issue, we have developed ModchelinGuide, a ranking and improvement suggestion system for restaurants designed to help restaurant owners identify the key areas for improvement and view their performance rankings compared to nearby surrounding restaurants, accessible through our website. This system is targeted at restaurant owners, focusing on restaurants in the vicinity of King Mongkut's University of Technology Thonburi, including those located along Soi Anamai Ngam Charoen 25, Pracha Uthit Road, Phutthabucha Road Soi 1 to 48, and Rama II Road.

The system is developed using a customer review dataset, with a machine learning model applied to generate improvement suggestions and rank restaurants. This report covers the entire development process, including data preprocessing, exploratory data analysis (EDA), methodology, analysis, results, and conclusion.

Chapter 1 Preprocessing

Preprocessing plays a crucial role in preparing raw data for accurate and efficient modeling. This chapter describes the preprocessing steps applied to transform raw data into a clean, reliable format, ready for further analysis and modeling.

1.1 Data Collection

1.1.1 Sources of Data

The customer review data used in this project was collected primarily from Google Maps through two distinct methodologies: multiple Google Maps APIs and web scraping. The goal was to gather comprehensive Thai and English restaurant review data in areas surrounding four main streets in Bangmod:

- Pracha Uthit Road
- Phutthabucha Road
- Anamai Ngam Charoen Road
- Rama II Road

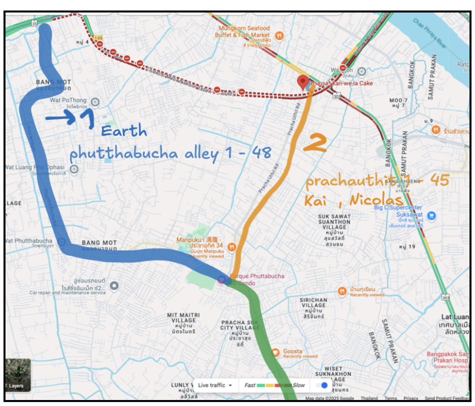

Figure 1: Phutthabucha alley 1 - 48 and Prachauthit alley 1- 45

Figure 2: Anamai Ngam Charoen Road

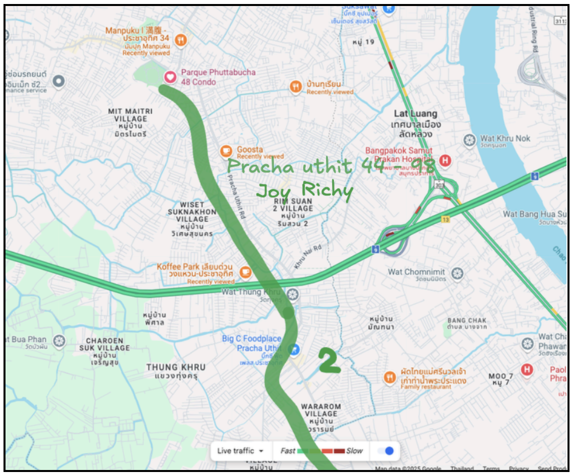

Figure 3: Prachauthit alley 44 - 98

Figure 4: Rama II Road

These locations were specifically chosen due to their density and diversity of restaurant establishments, enabling a representative analysis of local dining preferences and sentiments.

1.1.2 Data Collection Methods

A) Google Maps APIs

The first approach employed Google Maps APIs to systematically retrieve structured data on restaurants and cafés within a defined radius (1,000 meters and more) around selected geographic coordinates (latitude and longitude). Key information collected through the APIs includes:

- Place ID (place_id)
- Restaurant name
- Address
- Phone number
- Website (if available)
- Overall rating
- User review counts

- Individual user reviews (text)

A Python script leveraging the API request library was created to automate the API calls and handle JSON responses. To ensure efficiency and performance, parallel processing (ThreadPoolExecutor) was used, enabling the simultaneous retrieval of multiple restaurant details. Results were systematically stored in CSV format after deduplication based on unique identifiers.

B) Web Scraping (Playwright)

Due to limitations in the number of reviews retrieved via Google's APIs, five reviews per restaurant, additional data collection was conducted through web scraping using the Playwright library. This method allowed the extraction of a larger volume of detailed user-written reviews directly from Google Maps web pages.

For each targeted restaurant, identified by its place ID, from the initial API results, the automated script performed the following tasks:

- Navigated to the specific Google Maps URL corresponding to each place ID.
- Clicked on an available button, such as "รีวิวเพิ่มเติม" or "Reviews", to expand the full reviews pane.
- Iteratively scrolled through dynamically loaded content to extract detailed review texts, user ratings, usernames, and review identifiers (review button).
- Extracted rating score by parsing the displayed star ratings.

The script incorporated randomized delays (5 to 10 seconds) between page requests to ensure polite, responsible web scraping and to mitigate the risk of IP blocking.

All scraped data was recorded and continuously updated in a master CSV file. Progress was tracked through incremental storage to allow restarting the scraping process efficiently if interrupted.

1.1.3 Data Summary

The final combined dataset consisted of detailed restaurant reviews obtained from both the Google Maps APIs and web scraping, totaling around 70 thousand user-generated review records. After preliminary deduplication, removal of irrelevant phrases (for example, generic thank-you notes such as "ขอบพระคุณ"), and basic cleansing, the finalized data was stored as a CSV file.

The comprehensive approach ensured a robust and representative dataset suitable for subsequent preprocessing, exploratory data analysis, and modeling tasks in this project.

Road Name	**Data Source**	**Total Restaurants Collected**	**Total Thai Reviews Collected**	**Total English Reviews Collected**	**Total Review**
Pracha Uthit Road	Google Maps	425	23,116	9,157	31,125
Phutthabucha Road	Google Maps	123	9,405	10,050	19,455
Anamai Ngam Charoen Road	Google Maps	46	6,871	6,624	13,495
Rama II Road	Google Maps	57	3,668	3,258	6,926
TOTAL	-	651	43,060	29,089	**72,149**

1.2 Data Augmentation

After collecting the raw review dataset, a Cross-Lingual Data Augmentation process was performed to expand the dataset and introduce more varied text. This approach included translating reviews from the original Thai (TH) language into English (EN), paraphrasing the English version (rewriting the sentences while maintaining their original meaning), and then retranslating the paraphrased text back into Thai. Specifically, the process involved these steps:

1. Thai reviews → English translation

2. English text → English paraphrasing

3. Paraphrased English text → Thai translation

To measure the differences between the original and augmented reviews, the Hamming Distance (HD) score was calculated. This score was used to confirm that augmented reviews provided meaningful variation while preserving the original context and sentiment.

$$\mathrm{HD} = \frac{2(1-\cos(t_{\mathrm{org}}, t_{\mathrm{en}}))(1-\cos(t_{\mathrm{org}}, t_{\mathrm{th}}))}{(1-\cos(t_{\mathrm{org}}, t_{\mathrm{en}})) + (1-\cos(t_{\mathrm{org}}, t_{\mathrm{th}}))}$$

Figure 5: Hamming Distance score formula

This augmentation step significantly improved the diversity and quality of the dataset, preparing it effectively for the subsequent analysis and modeling stages.

1.3 Data Cleaning

Data Cleaning is one of the most important steps in our project. To ensure the reliability and usability of our dataset for analysis and modeling, we applied a thorough data cleaning strategy, focusing on structural and text-specific preprocessing. For example:

1.3.1 Handling Missing Data

We examined all fields for missing values. Records with missing review text were removed from the dataset, as the review text of the restaurant is essential for our sentiment, name entity recognition, and topic modeling tasks.

1.3.2 Outlier Treatment

Since our analysis centers on customer reviews, traditional numeric outlier detection was not applicable. However, we filtered out excessively short reviews (fewer than 3 words) as they often contained no meaningful content. These were considered textual outliers and removed from the dataset. For example, review text with only a single word without context, like อร่อย, อร่อยมาก, was removed.

1.3.3 Handling Duplicate Records

We identified and removed duplicate review entries by checking for exact matches in the review text. This ensured each review in the dataset was unique and contributed distinct information.

1.3.4 Text Preprocessing

In preparation for downstream NLP tasks, we applied the following text-specific cleaning steps:

- **Tokenization**: Split sentences into individual words for analysis
- **Stopword removal**: Eliminated common but non-informative words (“ครับ”, “ค่ะ”).
- **Symbol and emoji removal**: Stripped out all emojis, punctuation, and special characters to reduce noise.
- **Lowercasing**: Standardized all text to lowercase to treat similar words equally.

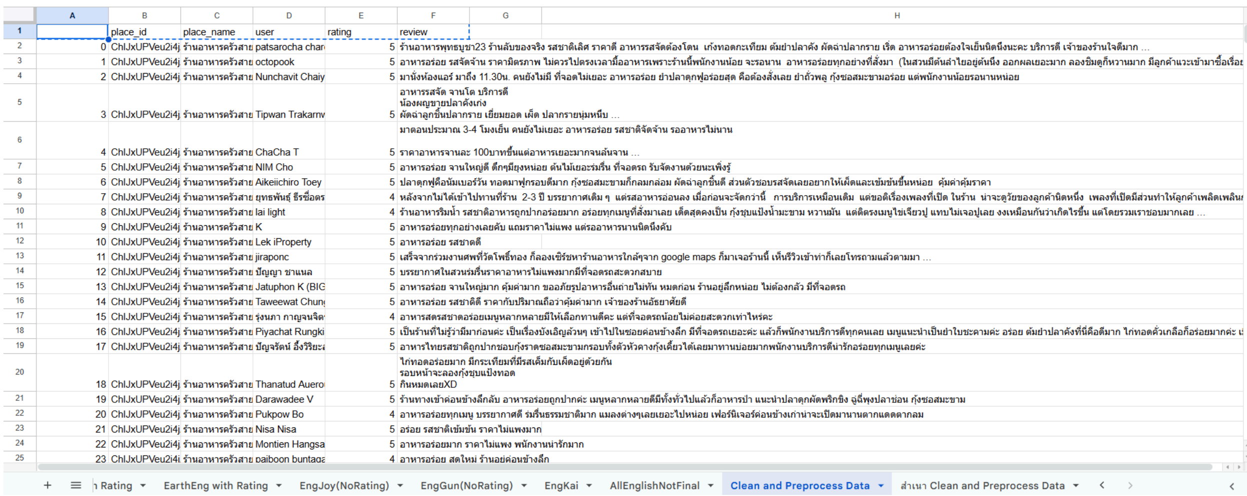

Figure 6: Image of cleaned Thai review dataset

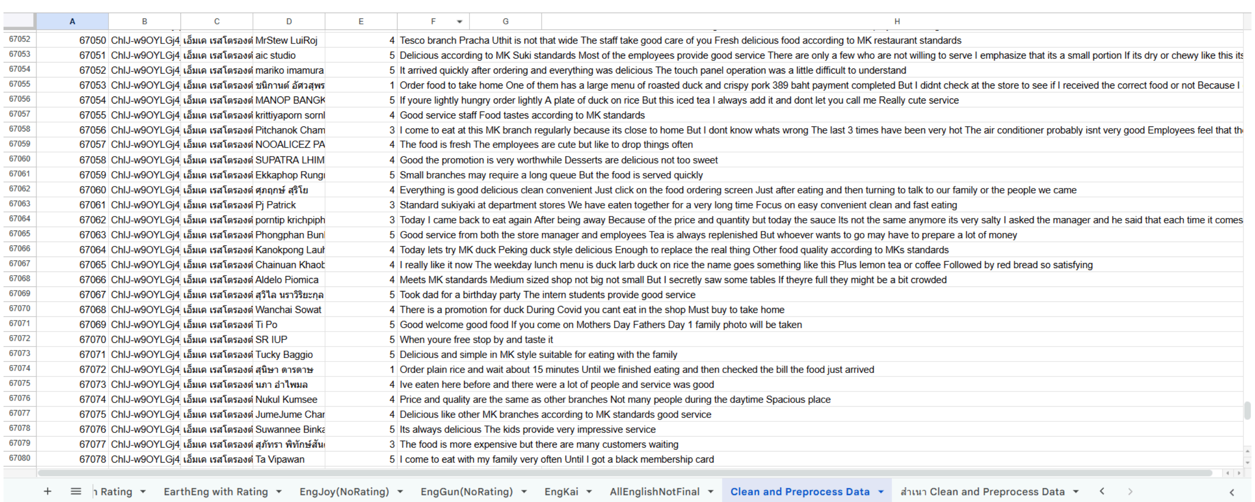

Figure 7: Image of cleaned English review dataset

These steps ensured our dataset was clean, consistent, and ready for machine learning modeling, particularly for topic modeling, Named Entity Recognition (NER), and Aspect-Based Sentiment Analysis (ABSA).

Chapter 2 Exploratory Data Analysis (EDA)

2.1 Topic Modeling

To do this project, we must first know what reviews on Google Maps are like, so we took a look and saw that there are patterns of what people often say to a restaurant in them. For example, whether the food is good or bad, whether the service is good or bad, and whether the place is clean or dirty. Still, for the sake of the credibility of the product, we did not want to assume what most people care about a lot from just what we saw, which was why we came up with using BERTopic to do topic modeling, so the aspects of reviews we would concern later on are statistical, not out of nowhere. The specifics of BERTopic will be discussed later on in the method section. Below is the image representation of all the topics we obtained from using BERTopic and their relations.

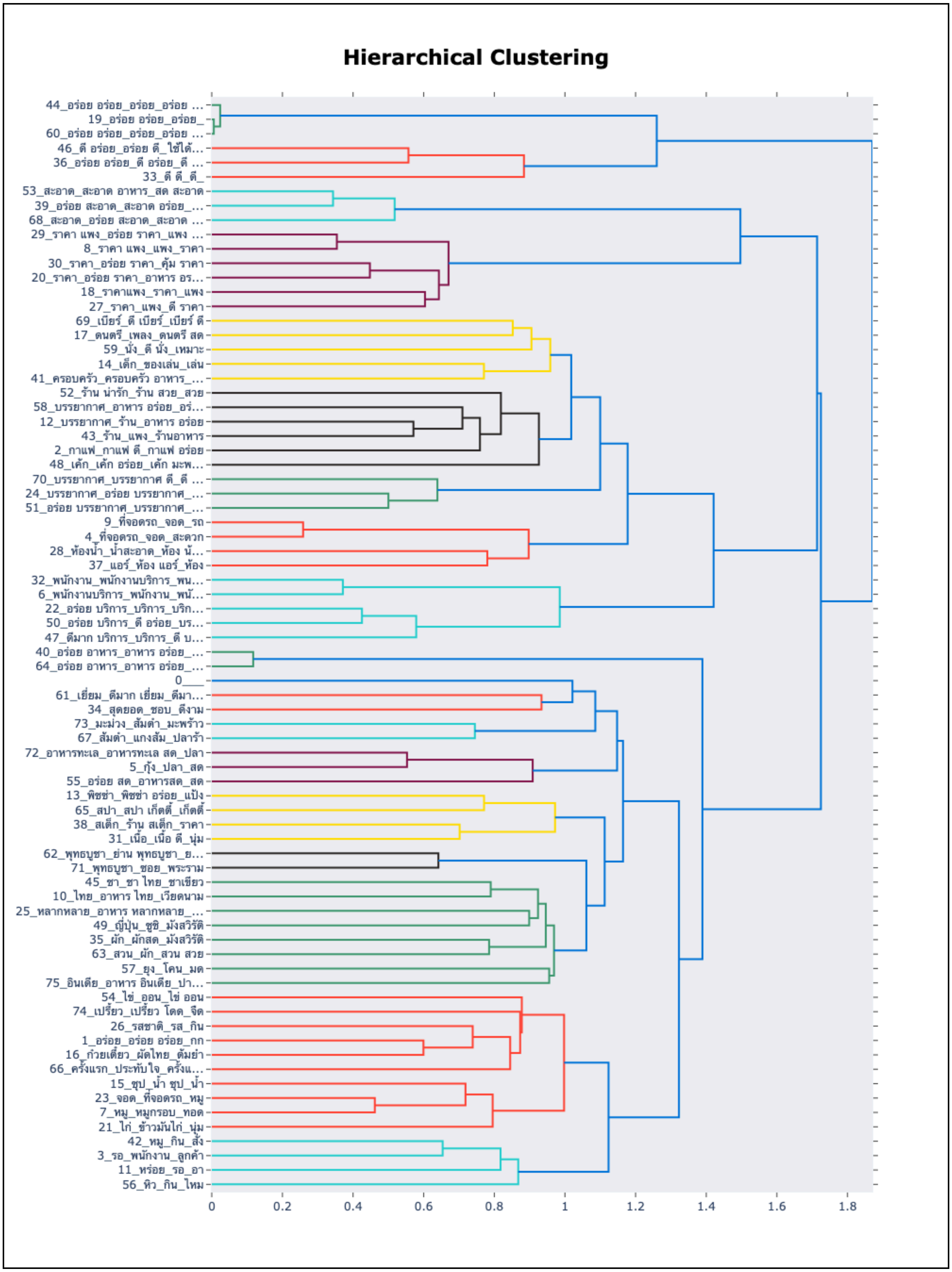

Figure 8: Hierarchical clustering visualization of the resulting topics from BERTopic

2.2 Named Entity Recognition (NER)

The image above reflects topics people normally talk about when reviewing restaurants in our selected areas. However, there is a lot of redundancy, and the wording is not yet human, which means it cannot be used right away, which is why we needed to do feature selection, meaning that we read all of those and extracted important keywords, which could be the higher-level word of multiple features combined. For instance, topics of beer, tea, and coffee are grouped into a single topic, drink.

After getting the actual important keywords, we needed to create a dataset for fine-tuning an NER model. Below is what it should look like.

BIO Tags Checkpoint.csv ×

1 to 10 of 5951 entries Filter

words	ner
['ร้าน', 'อาหาร', 'พุทธ', 'บูชา', '23', ' ', 'ร้าน', 'ลับ', 'ของ', 'จริง', ' ', 'รสชาติ', 'เลิศ', ' ', 'ราคา', 'ดี', ' ', 'อาหาร', 'รส', 'จัด', 'ต้อง', 'โดน', ' ', 'เก๋ง', 'ทอด', 'กะเทียม', ' ', 'ต้มยำ', 'ปลา', 'คัง', ' ', 'ผัด', 'ฉ่า', 'ปลา', 'กราย', ' ', 'เริ่ด', ' ', 'อาหาร', 'อร่อย', 'ต้อง', 'ใจ', 'เย็น', 'นิด', 'นึง', 'นะ', 'คะ', ' ', 'บริการ', 'ดี', ' ', 'เจ้า', 'ของ', 'ร้าน', 'ใจ', 'ดี', 'มาก', ' ', '...']	[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 13, 14, 0, 25, 26, 0, 15, 16, 16, 0, 0, 0, 15, 16, 16, 0, 15, 16, 16, 0, 15, 16, 16, 16, 0, 0, 0, 15, 16, 0, 0, 0, 0, 0, 0, 0, 0, 29, 30, 0, 0, 0, 0, 0, 30, 30, 0, 0]
['อาหาร', 'อร่อย', ' ', 'รส', 'จัดจ้าน', ' ', 'ราคา', 'มิตรภาพ', ' ', 'ไม่', 'ควร', 'ไป', 'ตรง', 'เวลา', 'มื้อ', 'อาหาร', 'เพราะ', 'ร้าน', 'นี้', 'พนักงาน', 'น้อย', ' ', 'จะ', 'รอ', 'นาน', ' ', 'อาหาร', 'อร่อย', 'ทุก', 'อย่าง', 'ที่', 'สั่ง', 'มา', ' ', '(', 'ใน', 'สวน', 'มี', 'ต้น', 'ลำไย', 'อยู่', 'ต้น', 'นึง', ' ', 'ออก', 'ผล', 'เยอะ', 'มาก', ' ', 'ลอง', 'ชิม', 'ดู', 'ก็', 'หวาน', 'มาก', ' ', 'มี', 'ลูกค้า', 'แวะ', 'เข้า', 'มา', 'ซื้อ', 'เรื่อยๆ', ' ', 'ด้วย', 'ค่ะ', ')']	[15, 16, 0, 13, 14, 0, 25, 26, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 29, 30, 0, 0, 35, 36, 0, 15, 16, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 37, 38, 0, 0, 0, 0, 0, 38, 38, 0, 0, 0, 0, 0, 38, 38, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
['มา', 'นั่ง', 'ห้อง', 'แอร์', ' ', 'มา', 'ถึง', '11.30', 'น.', ' ', 'คน', 'ยัง', 'ไม่', 'มี', ' ', 'ที่', 'จอด', 'ไม่', 'เยอะ', ' ', 'อาหาร', 'อร่อย', ' ', 'ยำ', 'ปลาดุก', 'ฟู', 'อร่อย', 'สุด', ' ', 'คือ', 'ต้อง', 'สั่ง', 'เลย', ' ', 'ยำ', 'ถั่ว', 'พลู', ' ', 'กุ้ง', 'ซอส', 'มะขาม', 'อร่อย', ' ', 'แต่', 'พนักงาน', 'น้อย', 'รอ', 'นาน', 'หน่อย']	[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 23, 24, 24, 24, 0, 15, 16, 0, 15, 16, 16, 16, 16, 0, 0, 0, 0, 0, 0, 15, 16, 16, 0, 15, 16, 16, 16, 0, 0, 29, 30, 35, 36, 36]
['อาหาร', 'รส', 'จัด', ' ', 'จาน', 'โต', ' ', 'บริการ', 'ดี', ' ', 'น้อง', 'ผญ', 'ขาย', 'ปลา', 'คัง', 'เก่ง', ' ', 'ผัด', 'ฉ่า', 'ลูกชิ้น', 'ปลา', 'กราย', ' ', 'เยี่ยมยอด', ' ', 'เผ็ด', ' ', 'ปลา', 'กราย', 'นุ่ม', 'หนึบ']	[15, 13, 14, 0, 31, 32, 0, 29, 30, 0, 0, 0, 0, 15, 16, 0, 0, 15, 16, 16, 16, 0, 0, 0, 0, 13, 0, 15, 16, 14, 14]
['มา', 'ตอน', 'ประมาณ', '3-4', 'โมง', 'เย็น', ' ', 'คน', 'ยัง', 'ไม่', 'เยอะ', ' ', 'อาหาร', 'อร่อย', ' ', 'รสชาติ', 'จัดจ้าน', ' ', 'รอ', 'อาหาร', 'ไม่', 'นาน', ' ', 'ราคา', 'อาหาร', 'จาน', 'ละ', '100', 'บาท', 'ขึ้น', 'แต่', 'อาหาร', 'เยอะ', 'มาก', 'จน', 'ล้น', 'จาน', ' ', '...']	[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 15, 16, 0, 13, 14, 0, 35, 36, 36, 36, 0, 25, 26, 26, 26, 26, 26, 26, 0, 39, 40, 40, 40, 40, 40, 0, 0]
['อาหาร', 'อร่อย', ' ', 'จาน', 'ใหญ่', 'ดี', ' ', 'ตึก', 'ๆ', 'มี', 'ยุง', 'หน่อย', ' ', 'ต้นไม้', 'เยอะ', 'ร่มรื่น', ' ', 'ที่', 'จอด', 'รถ', ' ', 'รับ', 'จัด', 'งาน', 'ด้วย', 'นะ', 'เพิ่ง', 'รู้']	[15, 16, 0, 31, 32, 32, 0, 35, 36, 0, 19, 20, 0, 1, 2, 2, 0, 23, 24, 24, 0, 29, 30, 30, 0, 0, 0, 0]
['ปลาดุก', 'ฟู', 'คือ', 'นัมเบอร์', 'วัน', ' ', 'ทอด', 'มา', 'ฟู', 'กรอบ', 'ดี', 'มาก', ' ', 'กุ้ง', 'ซอส', 'มะขาม', 'ก็', 'กลมกล่อม', ' ', 'ผัดฉ่า', 'ลูกชิ้น', 'ดี', ' ', 'ส่วนตัว', 'ชอบ', 'รส', 'จัด', 'เลย', 'อยาก', 'ให้', 'เผ็ด', 'และ', 'เข้มข้น', 'ขึ้น', 'หน่อย', ' ', 'คุ้มค่า', 'คุ้ม', 'ราคา']	[15, 16, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 15, 16, 16, 0, 13, 0, 15, 16, 0, 0, 0, 0, 13, 14, 0, 0, 0, 14, 0, 14, 0, 0, 0, 25, 26, 26]
['หลังจาก', 'ไม่', 'ได้', 'เข้าไป', 'ทาน', 'ที่', 'ร้าน', ' ', '2-3', ' ', 'ปี', ' ', 'บรรยากาศ', 'เดิม', ' ๆ', ' ', 'แต่', 'รส', 'อาหาร', 'อ่อน', 'ลง', ' ', 'เมื่อก่อน', 'จะ', 'จัด', 'กว่า', 'นี้', ' ', 'การ', 'บริการ', 'เหมือน', 'เดิม', ' ', 'แต่', 'ขอ', 'ติ', 'เรื่อง', 'เพลง', 'ที่', 'เปิด', ' ', 'ใน', 'ร้าน', ' ', 'น่า', 'จะ', 'ดู', 'วัย', 'ของ', 'ลูกค้า', 'นิด', 'หนึ่ง', ' ', 'เพลง', 'ที่', 'เปิด', 'มี', 'ส่วน', 'ทำให้', 'ลูกค้า', 'เพลิดเพลิน', 'กับ', 'อาหาร', ' ', 'หรือ', ' ', 'กิน', 'ไม่', 'ลง', 'ก็', 'ได้', ' ', 'อยาก', 'จะ', 'ให้', 'ร้าน', 'ศึกษา', 'เรื่อง', 'นี้', 'ด้วย', 'ครับ']	[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 2, 2, 0, 0, 13, 14, 14, 14, 0, 0, 0, 0, 0, 0, 0, 29, 30, 30, 30, 0, 0, 0, 0, 11, 12, 12, 12, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 11, 12, 12, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
['ร้าน', 'อาหาร', 'ริม', 'น้ำ', ' ', 'รสชาติ', 'อาหาร', 'ถูก', 'ปาก', 'อร่อย', 'มาก', ' ', 'อร่อย', 'ทุก', 'เมนู', 'ที่', 'สั่ง', 'มา', 'เลย', ' ', 'เด็ด', 'สุด', 'คง', 'เป็น', ' ', 'กุ้ง', 'ชุบ', 'แป้ง', 'น้ำ', 'มะขาม', ' ', 'หวาน', 'มัน', ' ', 'แต่', 'ติ', 'ตรง', 'เมนู', 'ไข่', 'เจียว', 'ปู', ' ', 'แทบ', 'ไม่', 'เจอ', 'ปู', 'เลย', ' ', 'งง', 'เหมือน', 'กัน', 'ว่า', 'เกิด', 'ไร', 'ขึ้น', ' ', 'แต่', 'โดย', 'รวม', 'เรา', 'ชอบ', 'มาก', 'เลย', ' ', '...']	[0, 0, 0, 0, 0, 13, 14, 14, 14, 14, 14, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 15, 16, 16, 16, 16, 0, 14, 14, 0, 0, 0, 0, 15, 16, 16, 16, 0, 0, 0, 0, 16, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
['อาหาร', 'อร่อย', 'ทุก', 'อย่าง', 'เลย', 'คับ', ' ', 'แถม', 'ราคา', 'ไม่', 'แพง', ' ', 'แต่', 'รอ', 'อาหาร', 'นาน', 'นิด', 'นึง', 'คับ']	[15, 16, 0, 0, 0, 0, 0, 0, 25, 26, 26, 0, 0, 35, 36, 36, 36, 36, 0]

Show 10 per page 1 2 10 100 500 590 596

Figure 9: A dataset for fine-tuning the NER model

For each row, the words column contains a list of words from the review, and the NER column contains the numerical representation of the BIO tag of the corresponding word. For example, suppose we assign 15 to B-Atmosphere, and a word has the corresponding number of 15, its BIO tag is B-Atmosphere. The specifics of BIO tags will be explained later on in the method section.

The technique we used is to use PyThaiNLP to separate words in a review, tokenize them, and for each one, compute its cosine similarity score with each category above, and it will be tagged with the category with the highest score with the help of additional algorithm to determine B and I. Note that words with the highest score below a threshold, 0.5, will immediately be tagged with O. The result from using this method was very poor, which means there were a lot of words that we intended to be in one category when it ended up in another. To resolve, we change some category names to their synonyms, which helps only a bit.

We tried to address this further by creating a small list of example words for each category, tokenizing them, finding the mean of each category, and using that to compare the cosine similarity score with the words in the review instead. Some categories were better, and some were worse, so we decided to use this method only for the categories that were not good when we were using the first method. Although the result was better, it is still not feasible.

The last solution we tried was to compare each word in the review with each word in the word list of each category directly and choose the maximum score. The result was significantly better, even if a new problem sprang. Words that are supposed to have O tags now either have B or I tags. We observed that those words only have tags of categories that have lists of example words, which is why we adjusted the algorithm so that only those categories have a higher threshold, which if it is not met, the word will gain the tag of one of the categories with a lower threshold instead. Even though the result was considerably better and was ready to be used further, we wanted it to be perfect, so we tried to come up with something new.

What we came up with was to use an API from Grok, a large language model, to do the work for us. The result was finally perfect, and the specifics will be in the method section.

Chapter 3 Method

3.1 Topic Modeling

Topic modeling was performed to discover common themes in the restaurant reviews. The BERTopic method was chosen for this task because it supports multilingual text, making it ideal for analyzing Thai reviews along with their augmented versions. Reviews were first converted into embeddings (numerical representations of meaning) using the *sentence-transformers model "paraphrase-multilingual-mpnet-base-v2.*

These embeddings allowed reviews to be clustered into groups, each representing a common topic or theme in the reviews. For each cluster, the top keywords (words or phrases most representative of that topic) were extracted. Then, these keywords were used as clear, concise labels for the topics. This helped simplify the analysis and provide meaningful insights about what aspects (like food, service, or pricing) customers commonly discussed.

The resulting topics provided the basis for further steps such as identifying key aspects, sentiment analysis, and predicting restaurant rating score. The result was represented in clusters.

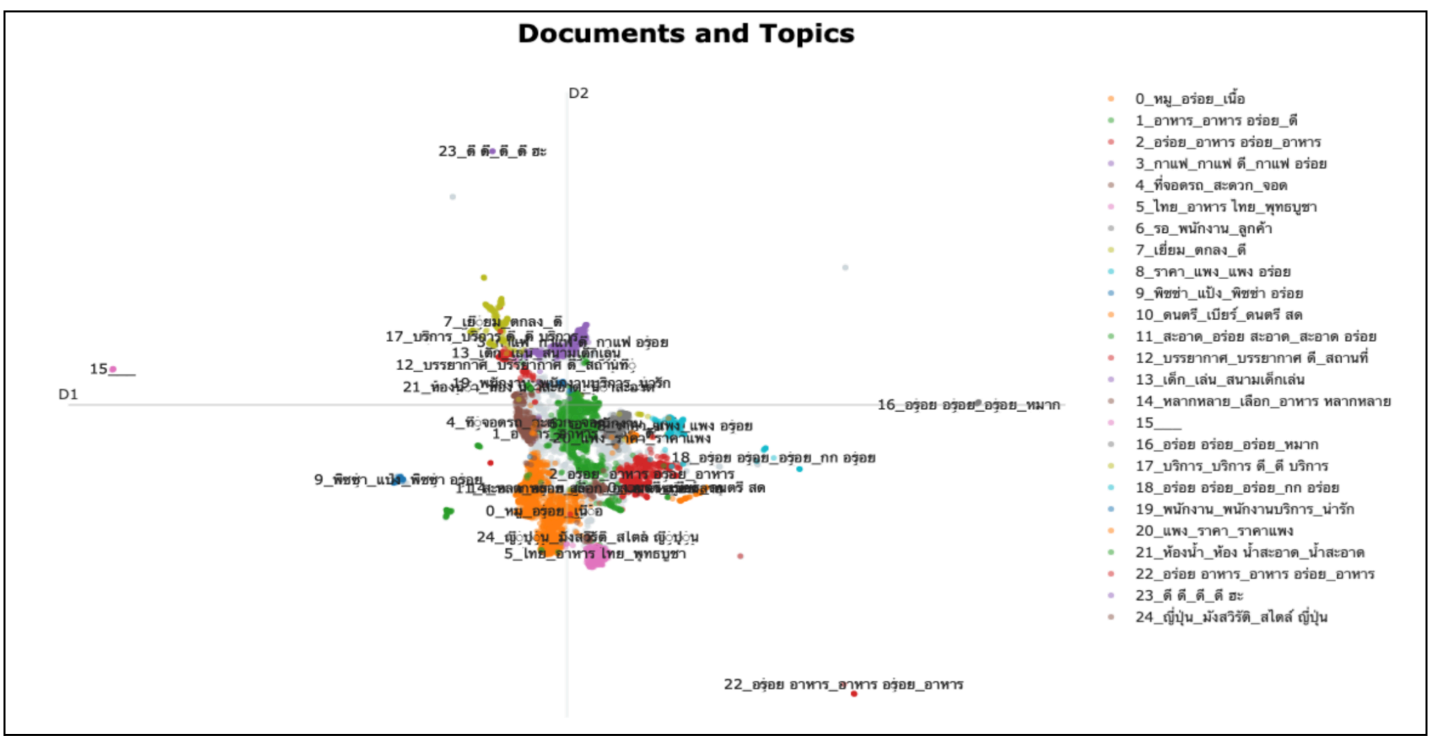

Figure 10: Visualization of the resulting topics generated from BERTopic

3.2 BIO-Tagging

A BIO-tagging scheme is needed to prepare data for the aspect-based sentiment analysis (ABSA) model and the named entity recognition (NER) model, since these models require precise identification and classification of entities like location, price, and food within text to learn. This scheme uses B, I, and O to annotate each token in a sentence from our dataset based on its position within an entity belonging to its relevant category. B-Category indicates the beginning of an entity from the category, I-Category represents tokens inside the entity from the category, and O-Category denotes tokens outside of any categories.

3.2.1 Process

Our method for adding BIO tags is to use an API from Grok, which is a large language model, which enables us to write a Python notebook to send it a prompt along with our review data, categories we have (B-Flavor, I-Flavor, B-FoodItem, I-FoodItem, etc.), and examples of the output we want (‘อาหาร’: ‘B-FoodItem’, ‘อร่อย’: I-FoodItem, ‘บรรยากาศ’: ‘B-Atmosphere’, ‘ร่มรื่น’: ‘I-Atmosphere’, etc.). After that, we map the labels to their assigned number to turn the data into what should be used to fine-tune an NER model (‘B-Flavor’: 17, ‘I-Flavor’: 18, ‘B-FoodItem’: 15, ‘I-FoodItem’: 16, etc.). We optimized the code using asyncio, which makes it run like we are running it from five different notebooks in parallel, each with four batches per API request. The results from this process are shown in Figure 9.

3.3 Fine-Tuning a Named Entity Recognition (NER) Model

3.3.1 Model Selection

After preparing the dataset with BIO-Tagging, we fine-tuned the WangchanBERTa model with our dataset, as it is a pre-trained Thai language model that can be adapted to recognize the custom entity categories. The model was trained to identify specific entity categories relevant to restaurant reviews, e.g., food, price, restroom, and more.

3.3.2 Training Configuration and Evaluation Metrics

Training was performed over 20 epochs with a batch size of 64, and aimed to find the longest word for the tokenization. The dataset was split into 70% training, 20% validation, and 10% testing. The model performance was evaluated on metrics: training loss, validation loss, precision, F1 score, recall, and token-level accuracy.

3.4 Aspect-Based Sentiment Analysis (ABSA)

In order to better understand what customers feel about specific parts of their dining experience, we used a method called **Aspect-Based Sentiment Analysis (ABSA)**. This method does not just look at whether an entire review text is positive or negative overall but it breaks the review down into specific topics (called *aspects*) like *food*, *service*, *taste*, *price*, and more, and then figures out how the customer feels about each one.

Aspect Identification

We used a fine-tuned **Named Entity Recognition (NER)** model to find the important aspects in each review sentence. For example, in a review that says "The food was delicious but the service was horrible," the model identified two aspects:

- **Food**
- **Service**

Sentiment Classification

After identifying the aspects, we sent them to the **Grok API**, which looked at how each aspect is talked about in the sentence and decided whether the sentiment is **positive**, **negative**, or **neutral**. In the example above:

- **Food** is classified as **positive**
- **Service** is classified as **negative**

Ordinal Information

In addition to sentiment, the Grok API also provides an **ordinal** value, which tells us how many times the same aspect appears in a single review. This is useful when a customer repeats a point for emphasis. However, in most cases, the ordinal value is **0**, meaning the aspect is only mentioned once.

3.5 Rating Score Prediction Model

The goal of this component is to predict each restaurant's overall star rating on the scale of zero to five to rank it along with the surrounding restaurants.

Data and Feature Engineering

1. ABSA Output
    - Input: 450 simulated reviews across 5 restaurants, each review annotated with aspect (taste, price, service, ambience) and polarity (positive, negative, neutral).
    - Numeric Mapping: sent_score = +1 for positive, -1 for negative, 0 for neutral.

2. Aggregated Metrics (per restaurant, per aspect)

Feature	Formula	Interpretation
cnt_<aspect>	count of mentions	Volume of discussion → signal strength
ssum_<aspect>	sum of sent_score	Net sentiment bias (+ praise, – complaints)
<aspect>_mean	ssum_<aspect> ÷ cnt_<aspect> (or 0 if none)	Average tone, scaled -1 to +1

In the feature engineering stage, we compute three metrics for each of the four core aspects—taste, price, service, and ambience—for every restaurant. First, cnt_<aspect> records how many times that aspect was mentioned, reflecting its prominence in customer feedback. Next, ssum_<aspect> sums the sentiment scores, yielding a net positivity or negativity for the aspect. Finally, we derive <aspect>_mean, the average sentiment score, by dividing the net sum by the mention count (or zero when there are no mentions). Together, these twelve aspect-level features form the input matrix X, while the true star ratings (on a 0–5 scale) constitute the target vector y.

Model Training and Selection

Model training proceeds with an 80-20 split between training and hold-out test sets, using a reproducible random seed. During training, we employ 5-fold cross-validation on the training portion to compare several algorithms which are Linear Regression serves as a baseline, Ridge Regression adds L2 regularization to stabilize estimates, ElasticNet combines L1 and L2 penalties for sparse feature selection, and two tree-based methods which is Gradient Boosting and Random Forest that capture non-linear patterns and feature interactions. For larger datasets or advanced ranking tasks, one might also include XGBoost or LightGBM, but in our experiments on the simulated five-restaurant set, Random Forest achieved the lowest mean absolute error (MAE) on the test fold.

Evaluation Metrics

We assess model performance primarily via MAE, which measures the average absolute deviation (in "stars") between predictions and ground truth, and R^2, which indicates the proportion of rating variance explained. An MAE below 0.5 star is generally acceptable for a 0–5 scale, while R^2 values closer to one denote strong explanatory power.

3.6 Rating Score Prediction Model

The Rating Score Prediction Model aims to translate fine-grained, aspect-level sentiments extracted from customer reviews into an accurate prediction of each restaurant's overall star rating. We begin by merging the ABSA output—each review's identified categories or aspects (‘price', 'fooditem', 'service', 'flavor', 'waiting', 'atmosphere', 'parkingspace', 'size', 'entertainment', 'fresh', 'childfriendly', 'insect', 'drink', 'portion', 'vegetable', 'cleanliness', 'country', 'restroom', 'meat', 'fruit') and its associated sentiment with the original review datase. This ensures that every sentiment annotation is correctly tagged with its restaurant's identifier and corresponding star rating.

Once merged, we convert the categorical sentiment labels into a numeric score (+1 for positive, -1 for negative, and 0 for neutral). To capture both the volume and tone of feedback, we aggregate these scores at the restaurant and category level. For each category we compute 4 primary statistics: the total mention count (cnt_<category>), the net sentiment sum (ssum_<category>), the raw counts of positive and negative mentions (pos_cnt_<category>, neg_cnt_<category>), and, to stabilize skewed distributions, the logarithm of the mention count (logcnt_<category>). From these, we derive three additional features: the average sentiment per mention (<category>_mean), the ratio of positive mentions (<category>_posratio), and the ratio of negative mentions (<category>_negratio). This rich feature set—over twenty metrics per restaurant—forms the input matrix X, while each restaurant's true average star rating becomes our target vector y.

place_id	cnt_atmosphere	cnt_childfriendly	cnt_cleanliness	cnt_country	cnt_drink	cnt_entertainment	cnt_flavo
ChIJ-1Pq0gmj4jARt7GoJPdD59M	10	0	0	4	1	0	[illegible]
ChIJ-QGexQSj4jARc8nFd9uTDMA	0	0	0	0	0	0	[illegible]
ChIJ-Vojge6i4jARjGOaUl1NVo0	25	8	1	1	4	2	[illegible]
ChIJ-ddrbRCj4jARnFNPhtmppCE	8	0	4	2	3	0	[illegible]
ChIJ0SU6DE6l4jARrn94uznljjY	0	0	1	0	0	0	[illegible]

Figure 11.1: Total mentioned count of each category in each restaurant (Part 1)

t_fooditem	cnt_fresh	cnt_fruit	...	restroom_negratio	restroom_logcnt	meat_mean	meat_posratio	meat_negratio	meat_logcnt	frui
16	0	0	...	0.0	0.0	0.5	0.5	0.0	1.098612	
0	0	0	...	0.0	0.0	0.0	0.0	0.0	0.000000	
72	2	1	...	0.0	0.0	0.0	0.0	0.0	0.000000	
27	1	0	...	0.0	0.0	1.0	1.0	0.0	1.609438	
2	0	0	...	0.0	0.0	0.0	0.0	0.0	0.000000	

Figure 11.2: Total mentioned count of each category in each restaurant (Part 2)

To identify the most effective prediction algorithm, we evaluated three state-of-the-art regressors—ElasticNet (combining L1 and L2 regularization), Random Forest, and XGBoost—using a 5-fold cross-validation strategy. We measured performance by Mean Absolute Error (MAE) and R^2, ensuring models were compared fairly on out-of-fold predictions only.

Chapter 4 Analysis

4.1 Fine-Tuned Named Entity Recognition (NER) Model

4.1.1 Performance Evaluation

The fine-tuning of the WangchanBERTa model[2] shows improvement over 20 training epochs. The model has an average loss of 0.098 over 3,720 batches during training. By the final epoch (epoch 20), the model also had a training loss of 0.098 and a validation loss of 0.462. The precision, recall, and F1 score were 0.72, 0.75, and 0.73, respectively, with token-level accuracy reaching approximately 89.9%.

While a gap between training and validation loss is concerning and suggests that the model is mildly overfitting, the model demonstrates strong performance and can generalize to unseen data well, as evidenced by its high precision, recall, F1 score, and token-level accuracy, which are all higher than 0.7.

Epoch	Training Loss	Validation Loss	Precision	Recall	F1	Accuracy
1	No log	0.412199	0.683642	0.730067	0.706092	0.887275
2	No log	0.391448	0.683074	0.738958	0.709918	0.887750
3	0.138700	0.380073	0.708390	0.735458	0.721671	0.894632
4	0.138700	0.384399	0.706207	0.739336	0.722392	0.893244
5	0.138700	0.390188	0.714494	0.742741	0.728344	0.896115
6	0.132800	0.393977	0.719596	0.747092	0.733086	0.897882
7	0.132800	0.402805	0.713127	0.748132	0.730210	0.896647
8	0.132800	0.414257	0.712703	0.748227	0.730033	0.895639
9	0.110600	0.414367	0.722263	0.748700	0.735244	0.899213
10	0.110600	0.425905	0.718083	0.751159	0.734249	0.897673
11	0.093100	0.430443	0.724668	0.748794	0.736534	0.898034
12	0.093100	0.439849	0.717470	0.748889	0.732843	0.897122
13	0.093100	0.440566	0.717902	0.745673	0.731524	0.896742
14	0.083100	0.446141	0.721148	0.751159	0.735847	0.898034
15	0.083100	0.451700	0.719014	0.750024	0.734191	0.897312
16	0.083100	0.452370	0.723261	0.750213	0.736490	0.899289
17	0.075400	0.457533	0.720785	0.750780	0.735477	0.898149
18	0.075400	0.459409	0.721648	0.752294	0.736652	0.898377
19	0.069600	0.461756	0.719630	0.751348	0.735147	0.898225
20	0.069600	0.462204	0.719924	0.750497	0.734892	0.898643

Figure 12: The performance of the fine-tuned NER model

4.1.2 Output

We used the *newmm* (new maximal matching) engine from *Pythainlp,* a dictionary-based word segmentation engine that applies the maximal matching algorithm. This engine finds the longest possible word by scanning the text from left to right and matching it against a comprehensive Thai dictionary. We implemented the engine for sentence tokenization before performing the NER. For example, the sentence “อาหารอร่อยมากบริการดีเยี่ยมห้องน้ำสะอาด”, was tokenized into อาหาร, อร่อย, มาก, บริการ. ดีเยี่ยม, ห้อง, น้ำสะอาด.

These tokens were then applied with our fine-tuned NER model and obtained the output shown in Figure 12, which showed that the NER model could accurately label which tokens belong together as a single entity. For instance, the token "อาหาร" was labeled as B-FoodItem (beginning of a food item), while "อร่อย" and "มาก" were labeled as I-FoodItem (inside the same food item). This demonstrates that the model correctly interpreted "อาหารอร่อยมาก" as one continuous phrase rather than three unrelated words (which all tokens would have been labeled as B-FoodItem). By maintaining this connection, the model preserved the key meaning in communication, demonstrating strong performance in entity recognition.

```
อาหาร      → B-FoodItem
อร่อย      → I-FoodItem
มาก        → I-FoodItem
บริการ     → B-Service
ดีเยี่ยม    → I-Service
ห้อง       → B-Restroom
น้ำสะอาด    → I-Restroom
```

Figure 13: The output from the fine-tuned NER model

4.2 Review Sentiment Extraction

Grok was able to do a great job in extracting the sentiment out of each review, making the resulting dataset immediately ready to be used in the subsequent step. Below is an example of that dataset.

Sentiment.csv

1 to 100 of 37110 entries

text	line	length	category	aspect	sentiment	score
ร้านอาหารพุทธบูชา23 ร้านลับของจริง รสชาติเลิศ ราคาดี อาหารรสจัดต้องโดน เก้งทอดกะเทียม ต้มยำปลาคัง ผัดฉ่าปลากราย เริ่ด อาหารอร่อยต้องใจเย็นนิดนึงนะคะ บริการดี เจ้าของร้านใจดีมาก ...	1	179	price		positive	0
ร้านอาหารพุทธบูชา23 ร้านลับของจริง รสชาติเลิศ ราคาดี อาหารรสจัดต้องโดน เก้งทอดกะเทียม ต้มยำปลาคัง ผัดฉ่าปลากราย เริ่ด อาหารอร่อยต้องใจเย็นนิดนึงนะคะ บริการดี เจ้าของร้านใจดีมาก ...	1	179	price	ราคา ดี	positive	0
ร้านอาหารพุทธบูชา23 ร้านลับของจริง รสชาติเลิศ ราคาดี อาหารรสจัดต้องโดน เก้งทอดกะเทียม ต้มยำปลาคัง ผัดฉ่าปลากราย เริ่ด อาหารอร่อยต้องใจเย็นนิดนึงนะคะ บริการดี เจ้าของร้านใจดีมาก ...	1	179	fooditem		positive	0
ร้านอาหารพุทธบูชา23 ร้านลับของจริง รสชาติเลิศ ราคาดี อาหารรสจัดต้องโดน เก้งทอดกะเทียม ต้มยำปลาคัง ผัดฉ่าปลากราย เริ่ด อาหารอร่อยต้องใจเย็นนิดนึงนะคะ บริการดี เจ้าของร้านใจดีมาก ...	1	179	fooditem	เก้ง ทอด กะ เทียม	positive	0
ร้านอาหารพุทธบูชา23 ร้านลับของจริง รสชาติเลิศ ราคาดี อาหารรสจัดต้องโดน เก้งทอดกะเทียม ต้มยำปลาคัง ผัดฉ่าปลากราย เริ่ด อาหารอร่อยต้องใจเย็นนิดนึงนะคะ บริการดี เจ้าของร้านใจดีมาก ...	1	179	fooditem	ต้มยำ ปลา คัง	positive	0
ร้านอาหารพุทธบูชา23 ร้านลับของจริง รสชาติเลิศ ราคาดี อาหารรสจัดต้องโดน เก้งทอดกะเทียม ต้มยำปลาคัง ผัดฉ่าปลากราย เริ่ด อาหารอร่อยต้องใจเย็นนิดนึงนะคะ บริการดี เจ้าของร้านใจดีมาก ...	1	179	fooditem	ผัด ฉ่า ปลา ก ราย	positive	0
ร้านอาหารพุทธบูชา23 ร้านลับของจริง รสชาติเลิศ ราคาดี อาหารรสจัดต้องโดน เก้งทอดกะเทียม ต้มยำปลาคัง ผัดฉ่าปลากราย เริ่ด อาหารอร่อยต้องใจเย็นนิดนึงนะคะ บริการดี เจ้าของร้านใจดีมาก ...	1	179	service	เจ้าของ ร้าน ใจ ดี มาก	positive	0
อาหารอร่อย รสจัดจ้าน ราคามิตรภาพ ไม่ควรไปตรงเวลามื้ออาหารเพราะร้านนี้พนักงานน้อย จะรอนาน อาหารอร่อยทุกอย่างที่สั่งมา (ในสวนมีต้นลำไยอยู่ต้นนึง ออกผลเยอะมาก ลองชิมดูก็หวานมาก มีลูกค้าแวะเข้ามาซื้อเรื่อยๆ ด้วยค่ะ) ...	2	215	flavor	รส จัดจ้าน	positive	0
อาหารอร่อย รสจัดจ้าน ราคามิตรภาพ ไม่ควรไปตรงเวลามื้ออาหารเพราะร้านนี้พนักงานน้อย จะรอนาน อาหารอร่อยทุกอย่างที่สั่งมา (ในสวนมีต้นลำไยอยู่ต้นนึง ออกผลเยอะมาก ลองชิมดูก็หวานมาก มีลูกค้าแวะเข้ามาซื้อเรื่อยๆ ด้วยค่ะ) ...	2	215	price		positive	0
อาหารอร่อย รสจัดจ้าน ราคามิตรภาพ ไม่ควรไปตรงเวลามื้ออาหารเพราะร้านนี้พนักงานน้อย จะรอนาน อาหารอร่อยทุกอย่างที่สั่งมา (ในสวนมีต้นลำไยอยู่ต้นนึง ออกผลเยอะมาก ลองชิมดูก็หวานมาก มีลูกค้าแวะเข้ามาซื้อเรื่อยๆ ด้วยค่ะ) ...	2	215	price	ราคา มิตรภาพ	positive	0
อาหารอร่อย รสจัดจ้าน ราคามิตรภาพ ไม่ควรไปตรงเวลามื้ออาหารเพราะร้านนี้พนักงานน้อย จะรอนาน อาหารอร่อยทุกอย่างที่สั่งมา (ในสวนมีต้นลำไยอยู่ต้นนึง ออกผลเยอะมาก ลองชิมดูก็หวานมาก มีลูกค้าแวะเข้ามาซื้อเรื่อยๆ ด้วยค่ะ) ...	2	215	waiting	รอ นาน	negative	0
มานั่งห้องแอร์ มาถึง 11.30น. คนยังไม่มี ที่จอดไม่เยอะ อาหารอร่อย ยำปลาดุกฟูอร่อยสุด คือต้องสั่งเลย ยำถั่วพลู กุ้งซอสมะขามอร่อย แต่พนักงานน้อยรอนานหน่อย	3	151	atmosphere	ห้อง แอร์	positive	0
มานั่งห้องแอร์ มาถึง 11.30น. คนยังไม่มี ที่จอดไม่เยอะ อาหารอร่อย ยำปลาดุกฟูอร่อยสุด คือต้องสั่งเลย ยำถั่วพลู กุ้งซอสมะขามอร่อย แต่	3	151	parkingspace	ที่ จอด	neutral	0

Figure 14: The output from using Grok to extract sentiment

4.3 Regression Model Comparison

Model	MAE Mean	MAE Std	RMSE Mean	RMSE Std	R² Mean	R² Std
ElasticNet	0.258068	0.025483	0.369825	0.073344	0.391046	0.054060
RandomForest	0.224850	0.009229	0.320663	0.028075	0.517010	0.105332
XGBoost	0.231161	0.015176	0.340552	0.034275	0.434441	0.211355

Figure 15: Comparisons of the regression model performance

- Random Forest achieves the lowest average error (MAE ≈ 0.225) and the lowest RMSE (≈ 0.321), while explaining over half the variance (R^2 ≈ 0.517). Its fold-to-fold variability is also smallest (MAE σ ≈ 0.009), which means it generalizes most consistently across different slices of the data.

- XGBoost comes in a close second: MAE ≈ 0.231 and RMSE ≈ 0.341 , with R^2 ≈ 0.434. Its slightly higher standard deviations (MAE σ ≈ 0.015, R^2 σ ≈ 0.211) suggest it's more sensitive to the particular train/test split, but still captures substantial non-linear structure.

- ElasticNet (linear with L1+L2 regularization) trails behind, with MAE ≈ 0.258, RMSE ≈ 0.370, and R^2 ≈ 0.391. It shows moderate stability (MAE σ ≈ 0.025) but cannot match the tree-based models' ability to model complex interactions.

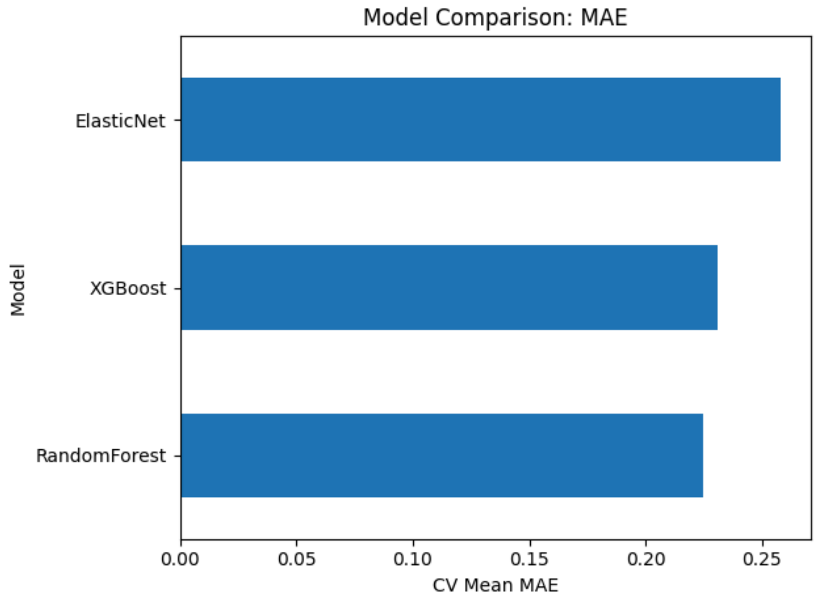

Figure 16: Model comparison based on MAE

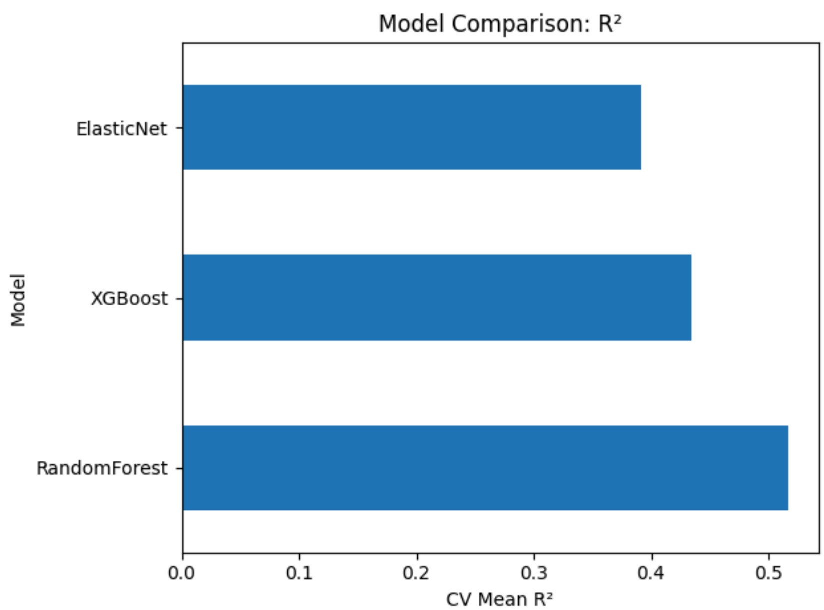

Figure 17: Model comparison based on R-squared

Random Forest not only yields the lowest prediction error but also the highest R^2 and the tightest fold-to-fold consistency, making it the preferred choice for our Rating Score Prediction task.

Chapter 5 Results & Discussion

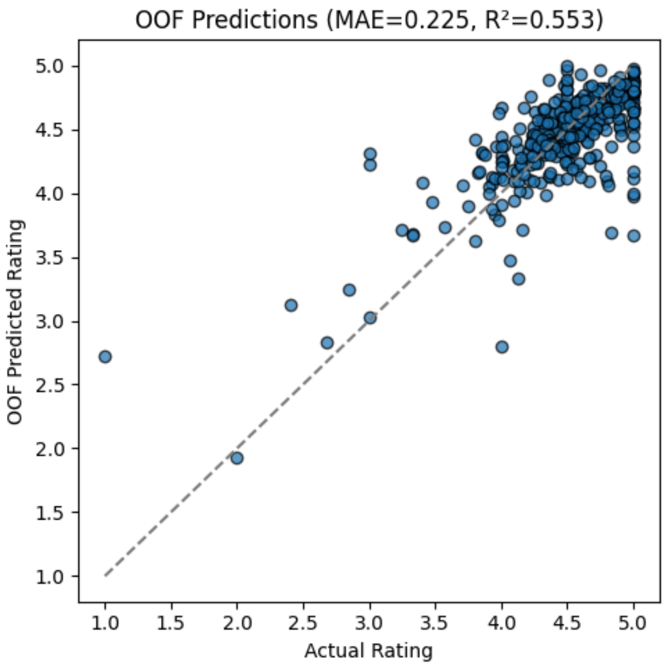

Figure 18: out-of-fold (OOF) predictions

The out-of-fold (OOF) results show that our chosen model predicts restaurant ratings with an average error of 0.225 stars (MAE) and explains about 55.3 % of the rating variance (R^2 = 0.553).

Looking at the OOF scatter plot, most points hug the diagonal line (especially in the 4.0–5.0 range), indicating the model generally gets high-rating restaurants right. A few outliers below the line reveal cases where it under-predicts, but those are relatively rare.

When we average feature importances across the folds, the negative-service ratio (service_negratio) towers above everything else at around 20 % importance. In other words, the more frequently diners complain about service (relative to how often they mention it at all), the more strongly the model lowers its star prediction.

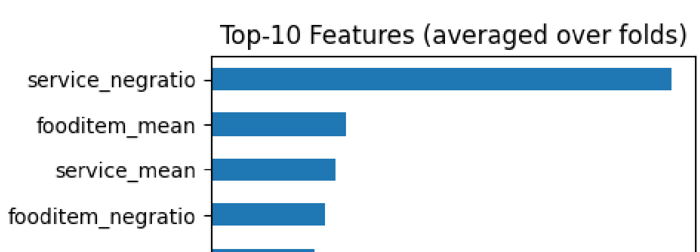

Figure 19: Comparisons of the mean importance of the top-10 features(averaged

After that, three food-related metrics each contribute roughly 5–6 %:

- fooditem_mean (average food sentiment)
- fooditem_negratio (fraction of negative food comments)
- fooditem_posratio (fraction of positive food comments)

A pair of price features also figures prominently (price_posratio, price_mean around 4 % each), showing that how often people praise or criticize the cost has a measurable effect on the overall rating.

Rounding out the top 10

- ssum_service (net service sentiment),
- cnt_atmosphere (raw count of atmosphere mentions), and
- atmosphere_logcnt (the log-transformed mention count).

To conclude, the result told us that

1. Service complaints drive down ratings most strongly—addressing negative service experiences should be the highest priority.

2. Food sentiment (both the average tone and the balance of positive versus negative comments) is the next key factor.

3. Price perceptions and atmosphere buzz also play important supporting roles.

With an MAE of just under a quarter-star and over half the variance explained, the model is reasonably accurate given the complexity of human opinions. To push further, we might incorporate richer text embeddings, temporal decay (weighting fresh reviews more), or additional metadata (cuisine type, location) to capture the remaining 45 % of unexplained variance.

Conclusion

We have successfully developed **ModchelinGuide**, a data-driven restaurant ranking and improvement suggestion system that leverages customer reviews to provide actionable insights. The system was built through a comprehensive pipeline that included data collection from Google Maps APIs and web scraping, resulting in a robust dataset of over 70,000 reviews. We applied a range of machine learning techniques, including topic modeling using BERTopic to uncover key aspects discussed by customers, fine-tuning a Named Entity Recognition (NER) model based on WangchanBERTa to extract meaningful entities, and implementing Aspect-Based Sentiment Analysis (ABSA) with the Grok API to evaluate sentiment toward specific aspects such as food, service, and cleanliness. A rating score prediction model was also developed, with Random Forest achieving the best performance in predicting restaurant ratings based on aspect-level sentiment data.

The system demonstrates several key strengths. It integrates multiple NLP components into a cohesive pipeline, provides fine-grained sentiment insights per aspect, and supports analysis in both Thai and English, making it well-suited for real-world applications in the Thai food service sector. Its use of machine learning enables it to go beyond general ratings and uncover specific strengths and weaknesses of each restaurant. However, there are also limitations. The current system is geographically constrained to the area of King Mongkut's University of Technology Thonburi and relies solely on Google Maps reviews, which may not reflect the full scope of customer feedback. Additionally, while the NER and ABSA components perform well, there is still room for improvement in handling other languages or rare entity categories.

Looking ahead, there are several opportunities to enhance the system. Expanding coverage to other districts or cities in Thailand would increase its impact and usability. Incorporating additional review platforms such as Wongnai, LINE MAN, or Facebook could provide a more comprehensive view of customer sentiment. Improvements to the NER model, such as using more advanced architectures or increasing training data diversity, could enhance accuracy. Furthermore, adding features like real-time dashboards, automatic review summarization, or interactive analytics would provide a richer user experience. These enhancements would further solidify ModchelinGuide as a valuable tool for restaurant owners seeking to understand and improve their business based on customer feedback.

References

[1] P. Pengpun, C. Udomcharoenchaikit, W. Buaphet, and P. Limkonchotiwat, "*Cross-Lingual Data Augmentation For Thai Question-Answering*," in Proceedings of the First Workshop on Generalization in the Age of Generative Language Models (GenBench 2023), 2023, [Online]. Available: https://aclanthology.org/2023.genbench-1.16.pdf

[2] AI Research at VISTEC, “wangchanberta-base-att-spm-uncased,” Hugging Face. [Online]. Available: https://huggingface.co/airesearch/wangchanberta-base-att-spm-uncased

[3] Sentence Transformers, “paraphrase-multilingual-mpnet-base-v2,” Hugging Face. [Online]. Available: https://huggingface.co/sentence-transformers/paraphrase-multilingual-mpnet-base-v2

[4] PyThaiNLP, “Thai Named Entity Recognition (NER) Version 2,” *PyThaiNLP*, [Online]. Available: https://pythainlp.org/Thai-NER/version/2.

Appendix

This is our model:
https://huggingface.co/wttw/modchelin_thainer-base-model